Semantic Business Process Regulatory Compliance Checking Using LegalRuleML

Guido Governatori, Mustafa Hashmi, Ho-Pun Lam, Serena Villata, Monica Palmirani

Semantic Business Process Regulatory Compliance Checking using LegalRuleML*

Guido Governatori[1], Mustafa Hashmi[1], Ho-Pun Lam[1],
Serena Villata[2] and Monica Palmirani[3]

[1] Data61, CSIRO, Spring Hill, QLD 4000, Australia
[2] Université Côte d'Azur, CNRS, Inria, I3S, France
[3] CIRSFID, University of Bologna

Abstract. Legal documents are the source of norms, guidelines, and rules that often feed into different applications. In this perspective, to foster the need of development and deployment of different applications, it is important to have a sufficiently expressive conceptual framework such that various heterogeneous aspects of norms can be modeled and reasoned with. In this paper, we investigate how to exploit Semantic Web technologies and languages, such as LegalRuleML, to model a legal document. We show how the semantic annotations can be used to empower a business process (regulatory) compliance system and discuss the challenges of adapting a semantic approach to legal domain.

1 Introduction

Business Process Management (BPM) is a set of methodologies to capture, model and control in an integrated way all those activities that take place in an environment defining an enterprise [6]. Companies are subject to regulations. Non-compliance to such regulations would not only affect the added-value of the business processes, but may also result in judiciary pursuits. The scope of norms is to regulate the behaviour of their subjects and to define what is legal and what is not [18]. In BPM, checking the *compliance* of a business process with respect to a set of relevant regulations means to identify whether a process violates or not a set of norms. Consequently, to ensure business processes are compliant we need two components: (i) a conceptually sound formal representation of a business process, and (ii) a conceptually sound formalism to model and reason with the norms derived by the regulations. The task of modelling legal norms requires substantial human effort and powerful languages to capture the semantics of the normative systems and their dynamics. This is one of the reasons why existing compliance frameworks [7, 21] are not fully satisfiable for companies.

We present an application of the semantic business process regulatory compliance checking where we relay on the semantics of LegalRuleML [1, 2] for the

* Supported by EU H2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 690974 for the project *MIREL: MIning and REasoning with Legal texts*.

representation of the norms and their dynamics. We discuss and analyse different but comparable ways to model the semantics of norms as well as their dynamics (e.g., new versions of certain regulations are proposed). Moreover, we show how this semantic modelling phase, with tasks coupled with semantic annotations, can be exploited to address and improve the regulatory compliance checking process, and answer companies' needs about compliance checking.

We experiment our approach on two versions of the Australian Telecommunications Consumer Protections Code[4] (hereafter, the Code). Our evaluation, in collaboration with an industry partner whose details cannot be disclosed for commercial reasons, shows that the proposed approach overcomes some of the drawbacks of standard non-semantic approaches to compliance checking.

The paper is organised as follows. In Section 2, we describe the context in which our approach has been conceived, answering the needs expressed by an industry partner. Section 3 describes how we model norms using LegalRuleML, and how we exploit the semantics of LegalRuleML to perform semantic regulatory compliance checking. In Section 4, we report on the results of the evaluation of the system, and discuss the insights inferred from this experience 4.2. Finally, we compare with the related literature in Section 5 and draw some conclusions.

2 Business Process Compliance

Regulatory compliance is a set of activities that aims to ensure that organisations' core business processes do not violate relevant regulations, in the jurisdiction in which the business is situated, governing the (industry) sectors where the organisation operates. Essentially, compliance connects two distinct domains: legal domain and business process domain.

Legal domain describes the legal boundaries for organisations by imposing conditions that detail which actions can be considered legal and which actions must be avoided during the execution of business process to stay compliant. Such legal boundaries can stem from normative documents (e.g., a code, bill, or an act) or organisation's internal policies (e.g., strategy documents or internal controls).

Business process domain, on the other hand, details how business activities should be carried out. A business process is a self-contained, temporally ordered set of activities describing how a process should be executed to achieve a business goal. Typically, it describes what needs to be done and when (control-flow), what resources is needed, who is/are involved (data and time), etc [17]. Many different formalisms (e.g., Petri-Nets. Process Algebra, ...) and notations (e.g., BPMN,YAWL, EPC, ...) have been proposed to present business processes. Apart from the differences in notations, typically a business process language is composed of the following minimal set of elements, namely: *tasks* (representing complex business activities), *connectors* (defining the relationships among the tasks of the process, i.e., sequence, AND-Join, AND-Split, XOR-Join, XOR-Split). The combination of tasks and connectors defines the possible ways in which a

[4] `http://www.commsalliance.com.au/Documents/all/codes/c628`

process can be executed—where a possible way, called *trace*, is a sequence of tasks executed by respecting the order given by the connectors.

However, compliance is not only about the tasks that an organisation has to perform to achieve its business objectives. It concerns also on their effects (i.e., how the activities in the tasks changes the environment in which they operate), and the artefacts produced by the tasks (e.g., the data resulting from executing a task or modified by the task) [17]. Hence, to check whether a business process complies with the relevant regulations, an annotated business process model and the formal representation of regulations is needed. Accordingly, Governatori and Sadiq [23] introduced the idea of *compliance-by-design* in which business processes are supplemented with additional information (by means of annotations representing the formalised regulations) to ensure that a business process is compliant with relevant normative frameworks before its actual deployment[5].

We report the results of a project in cooperation with an industry partner (a small-to-medium Australian Telco with 50K–100K customers) subject to the Code. The main objective is to exploit semantic technologies to empower the compliance-by-design methodology mentioned above. More precisely, our objectives are as follows:

- Model the regulatory code as well as its dynamics using a (machine-readable) semantic framework such that differences and connections between two versions of the code (namely, 2012 and 2016) can be automatically identified;
- Capturing the tasks, their effects, and the artefacts resulting from them by means of *semantic annotations*; and
- Extend the architecture of the *Regorous Process Designer*, a business process compliance checker based on the compliance-by-design approach proposed in [15], to account for the semantic annotations.

3 The Framework

We first present an overview of LegalRuleML (Sec. 3.1), and explain how it can be exploited to model the Code (Sec. 3.2). Finally, we describe the semantic annotations in LegalRuleML we associated to the tasks of the processes, and how they are used to address semantic regulatory compliance checking (Sec. 3.3).

3.1 LegalRuleML: An Overview

LegalRuleML[6] is an effort to create a standard for the representation of norms[7]. It builds on the experience of RuleML to provide a rule representation language and, at the same time, extends RuleML following the principles and guidelines

[5] There are other approaches to compliance checking, namely: run-time and post-execution approaches, see [18] for details.

[6] `https://tools.oasis-open.org/version-control/browse/wsvn/legalruleml/trunk/schemas/rdfs/modules/#_trunk_schemas_rdfs_modules`

[7] At the time of writing LegalRuleML is just about to enter in its public review phase.

proposed in [9] for rule language and legal reasoning. In particular, LegalRuleML offers facilities to model different types of norms (described below), deontic effects (e.g., obligations, prohibitions, permissions), and can specify preferences among them. In addition, it has features to capture the metadata of norms and other normative elements (such as jurisdiction, authorities, validity times, etc), and has mechanisms to implement the so called *legal isomorphism* [3] principle that establishes the connection between a legal source or a norm, and the corresponding formal representation.

Accordingly, a LegalRuleML document consists ot (Fig. 1): *metadata*, *statements* and *contexts*. The metadata part is meant to contain the legal sources of the norms modelled by the document, and information about the (legal) temporal properties of the sources and the document itself, the jurisdiction where the norms are valid, and eventually details describing the authorities, authors, ... for the legal sources and the document.

The statements part contains the formal representation of the norms in form of rules or other expressions supported by the language, as depicted in Fig. 2.

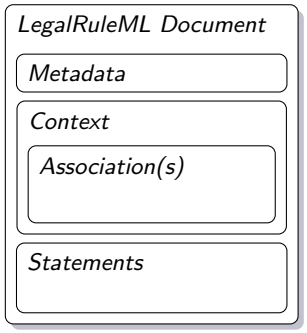

Fig. 1: LegalRuleML document structure.

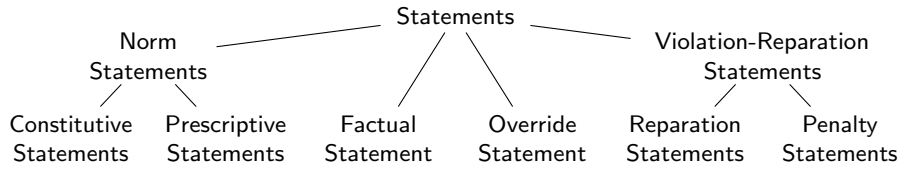

Fig. 2: Types of statements in LegalRuleML (adopted from [20]).

Normative statements follow the well known distinction of *constitutive* statements (rules) and *prescriptive* statements (rules) [24]. Constitutive rules are used to provide the definitions of the terms used in the document. For example, Chapter 2 of the Code provides the definitions of the terms used in the rest of Code. Often in legal documents, terms are defined defeasibly, thus the definition gives the base conditions that can be further extended or are subject to exceptions, e.g., "Complaint" is defined as:

> an expression of dissatisfaction made to a Supplier in relation to its Telecommunications Products or the complaints handling process itself, where a response or Resolution is explicitly or implicitly expected by the Consumer.
> An initial call to a provider to request a service or information or to request support is not necessarily a Complaint. An initial call to report a fault or

> service difficulty is not a Complaint. However, if a Customer advises that they want this initial call treated as a Complaint, the Supplier will also treat this initial call as a Complaint. If a Supplier is uncertain, a Supplier must ask a Customer if they wish to make a Complaint and must rely on the Customer's response.

Given the nature of the definition of the terms in legal reasoning, the definition of the terms can be captured by defeasible rules. In other terms, the constitutive rules provide the internal (defeasible) ontology used by the LegalRuleML document.

Prescriptive statements are rules that determine the deontic behaviour (such as obligations, prohibitions, permission) of the system and provide the conditions under which the deontic effects are in force.

Factual statements are meant to capture facts that are relevant in given cases. For instance, it can be used to specify that a particular manifestation of a norm (i.e., Section 8.4.1 of the 2012 version of the Code) is the same as another manifestation of the norm (i.e., Section 8.3.1 of the 2016 version of the Code).

Given that norms are represented by defeasible rules, and that two defeasible rules can be in conflict, override statements can be used to resolve the conflicts by specifying that, in case two rules in conflict fire at the same time, the stronger rule prevails over the weaker rule.

Finally, violation and reparation statements offer convenient ways to formalise the penalties that can potentially apply for breaches of norms and information about how the violated norms can be compensated.

3.2 Modelling the Code and its dynamics

The Telecommunication Consumer Protections Code is the Australian industry code for the telecommunication industry and mandates that every operator has to provide annual compliance statements with the Code. The Code was enacted in September 2012, and it entered in force in April 2013. In 2015, the Code was revised and a new version enacted with some amendments entered in force in 2016. In this paper, we consider the 2012 and 2016 versions of the Code, and model them using LegalRuleML[8].

The first (and simplest) option is to model two versions of the Code using two separate LegalRuleML documents. However, after we compare the two versions of the Code, we realise that while there are differences, the vast majority of the definitions and prescriptions are just the same. Thus, modelling with this option will result in a large among of duplicated statements (rules) with exactly the same structure and meaning.

The second option is to utilise LegalRuleML's features to link statements with their legal sources. To be able to do this, we have first create a set of statements covering all the rules that can be inherited from the Code, irrespective to which version of the Code the rules has been mentioned, as shown below[9]

[8] The data modelled in LegalRuleML is available at: `https://dl.dropboxusercontent.com/u/15116330/EKAW-dataset-legalruleml.zip`

[9] See Sec. 3.3 for how to model norms in LegalRuleML.

```
<lrml:Statements>
  <lrml:PrescriptiveStatement key="tcpc_ps1">
    ...
  </lrml:PrescriptiveStatement>
  <lrml:PrescriptiveStatement key="tcpc_ps2">
    ...
  </lrml:PrescriptiveStatement>
  <lrml:PrescriptiveStatement key="tcpc_ps3">
    ...
  </lrml:PrescriptiveStatement>
  ...
</lrml:Statements>
```

Next, we need to include the legal sources information about the two versions of the Code in the metadata section of the LegalRuleML document.

```
<lrml:LegalSources key="ls1">
  <lrml:LegalSource key="tcpc2012"
    sameAs="http://www.commsalliance.com.au/Documents/all/codes/c628#2012"/>
  <lrml:LegalSource key="tcpc2016"
    sameAs="http://www.commsalliance.com.au/Documents/all/codes/c628#2016"/>
</lrml:LegalSources>
```

Finally, we can make use of the `<lrml:Context>` element to create the associations between the legal sources and the statements, one for each version of the Code.

```
<lrml:Context key="tcpc2012-as">
  <lrml:Associations>
    <lrml:Association>
      <lrml:appliesSouce keyref="#tcpc2012/section8.2.1">
      <lrml:toTarget keyref="#tcpc_ps1">
    </lrml:Association>
    <lrml:Association>
      <lrml:appliesSouce keyref="#tcpc2012/section8.4.1">
      <lrml:toTarget keyref="tcps_ps2">
    </lrml:Association>
    ...
  </lrml:Associations>
<lrml:Context>

<lrml:Context key="tcpc2016-as">
  <lrml:Associations>
    <lrml:Association>
      <lrml:appliesSouce keyref="#tcpc2016/section8.2.1">
      <lrml:toTarget keyref="#tcpc_ps1">
    </lrml:Association>
    <lrml:Association>
      <lrml:appliesSouce keyref="#tcpc2016/section8.4.1">
      <lrml:toTarget keyref="#tcpc_ps3">
    </lrml:Association>
    ...
  </lrml:Associations>
</lrml:Context>
```

As can be seen from the snippet above, Section 8.2.1 is the same in both versions of the Code and is modelled by the same rule `tcpc_ps1`. On the other hand, Section 8.4.1 is different in the two versions of the Code. Thus the 2012 version is represented by the rule `tcpc_ps2`, while the 2016 by rule `tcpc_ps3`.

This option does not require us to duplicate the set of rules that are common among different versions of the legal document. The trade off is that we have to list all the associations for all the provisions in the Code and the corresponding rules. A more compact alternative would be to have a single association for each context with a single source (the entire version of the Code) and multiple

targets (the rules corresponding to that version). However, this alternative has the drawback to loose the semantic information about the relationship between the sections of the Code and corresponding rules (syntactically, the correspondence could be regained by establishing a schema for labelling the key of the rules).

3.3 Business process regulatory compliance

The set of traces T of a given business process describes the behaviour of the process insofar as it provides a description of all possible ways in which the process can be correctly executed. To check the semantic regulatory compliance of a process, we consider it as the set of its traces. The set of norms could vary from a particular regulation, to a specific statutory act, to a set of best practices, a standard, or simply a policy internal to an organisation or a combination of these types of prescriptive documents.

In this section, we provide an overview of the Regorous Process Designer, a business process regulatory compliance checker [16, 11], and how we extended it by enriching both the representation of norms and the business process tasks with the LegalRuleML semantic model. Starting from the norms modelled in LegalRuleML presented in Section 3.2, we need now to add such semantic annotations in LegalRuleML to the tasks of the process, using them to record the data, resources and other information related to the single tasks in a process.

For the formal representation of the regulation, Regorous use PCL (Process Compliance Logic) [10, 14]. PCL is a simple, efficient, flexible rule-based logic, obtained from the combination of defeasible logic (for the efficient treatment of exceptions which are quite common in normative reasoning), and a deontic logic of violations. In PCL, a norm is represented by a rule of the kind $a_1, \ldots, a_n \Rightarrow c$ where $a_1, \ldots, a_n$ are the conditions of applicability of the rule and c is the *normative effect* of the rule. PCL distinguishes two normative effects: the first is that of introducing a definition for a new term, e.g., the following rule from the Code (2012) specifies that if a Customer requests information about a Complaint, then it is deemed a consumer complaint activity.

$$complaint, requestInformation \Rightarrow consumerComplaintActivity$$

The second normative effect is that of triggering obligations and other deontic notions. For obligations and permissions, we use the following notations:

- [P]p: p is permitted;
- [OP]p: p is a punctual obligation;
- [OM]p: p is a maintenance obligation;
- [OAPP]p: p an achievement preemptive perdurant obligation;
- [OAPNP]p: p is an achievement preemptive non-perdurant obligation;
- [OANPP]p: p an achievement non preemptive perdurant obligation;
- [OANPNP]p: p is an achievement non preemptive non-perdurant obligation;
- [OM]$\neg p$: p is prohibited.

Rules involving obligations and permissions are a bit more complex. Let us consider the following example from the Code (Section 8.1.1.a.x.E): "The supplier must implement, operate and comply with a Complaint handling process that is transparent, including prohibiting a Supplier from cancelling a Consumer's Telecommunications Service only because, being unable to Resolve a Complaint with their Supplier, that Consumer pursued their options for external dispute resolution". This provision is translated into PCL in the following rule:

$$\neg resolution, complaint, externalDisputeResolution \Rightarrow [\mathrm{OM}]\neg terminateService.$$

This rule establishes that in case there is a complaint (*complaint*) that has not been resolved to the satisfaction to the consumer ($\neg$*resolution*) and the consumer opted for the external dispute resolution option (*externalDisputeResolution*), then the provider has the prohibition to terminate the service (*terminateService*). For full description of PCL and its features, see [10, 14].

The above rule is translated using the LegalRuleML semantic model as follows:

```
<lrml:PrescriptiveStatement key="ps_tcpc_8_1_1_a_x_E">
  <ruleml:Rule key="tcpc_8_1_1_a_x_E">
    <lrml:Paraphrase>The supplier must implement, operate and comply with a
      Complaint handling process that is transparent, including E. prohibiting
      a Supplier from canceling a Consumer's Telecommunications Service only
      because, being unable to Resolve a Complaint with their Supplier, that
      Consumer pursued their options for external dispute resolution.
    </lrml:Paraphrase>
    <lrml:hasStrength>
      <lrml:DefeasibleStrength
        iri="http://spin.nicta.com.au/spindle/ruleStrength#defeasible"/>
    </lrml:hasStrength>
    <ruleml:if>
      <ruleml:And>
        <ruleml:Neg>
          <ruleml:Atom>
            <ruleml:Rel>resolution</ruleml:Rel>
          </ruleml:Atom>
        </ruleml:Neg>
        <ruleml:Atom>
          <ruleml:Rel>complaint</ruleml:Rel>
        </ruleml:Atom>
        <ruleml:Atom>
          <ruleml:Rel>external dispute resolution</ruleml:Rel>
        </ruleml:Atom>
      </ruleml:And>
    </ruleml:if>
    <ruleml:then>
      <lrml:Obligation iri="http://test.org/deontic#OM">
        <ruleml:Neg>
          <ruleml:Atom>
            <ruleml:Rel>terminate Service</ruleml:Rel>
          </ruleml:Atom>
        </ruleml:Neg>
      </lrml:Obligation>
    </ruleml:then>
  </ruleml:Rule>
</lrml:PrescriptiveStatement>
```

Enriching the regulatory compliance system with a semantic representation of the regulations the processes have to be checked against presents many advantages, i.e., a more insightful and precise representation of the semantics of the norms, and the possibility to keep track of the regulations' dynamics. However, compliance

is not just about the tasks to be executed but also on what the tasks do, the way they change the data and the state of the artefacts related to the process, and the resources linked to it. Accordingly, process models must be enriched with such information [23]. For this reason, we decided to enrich process models with *semantic* annotations using the LegalRuleML model. Each task in a process model is then associated with a set of semantic annotations in LegalRuleML, representing the effects of the task. The approach can be used to model business process data compliance [17]. The set of effects in PCL and in LegalRuleML is just a set of literals in the underlying language. PCL and LegalRuleML are agnostic about the nature of the literals they uses. They can represent tasks, i.e., activities executed in a process, or propositions representing state variables.

An example of task annotated using LegalRuleML is the following: suppose that the complaint handling process of a telco contains a task called "Record Complaint". The Code (Section 8.5 of the 2012 version, and Section 8.4 of the 2016 version) specifies what information should be recoded for a complaint. Thus, the task "Record Compliant" indicates that such an activity is to be performed once a compliant as been verified as such, but, the process alone does not specify what data is recorded. Thus, such process model must be extended with the appropriate information. Note that it is beyond the scope of this paper to study how the annotations are generated, i.e., manually based on domain experts knowledge of the process or by examining database schemas associated to the task or programming script executed by the task [17]. Specifically, for the task "Record Complaint" the following literals (from the literals defined for the LegalRuleML document) are recorded as annotation for the task:

```
<taskEffects elementId="usertask15">
  <ruleml:Atom>
    <ruleml:Rel>record special circumstances</ruleml:Rel>
  </ruleml:Atom>
  <ruleml:Atom>
    <ruleml:Rel>record complaint issue</ruleml:Rel>
  </ruleml:Atom>
  <ruleml:Atom>
    <ruleml:Rel>record resolution sought</ruleml:Rel>
  </ruleml:Atom>
  <ruleml:Atom>
    <ruleml:Rel>record due date</ruleml:Rel>
  </ruleml:Atom>
  <ruleml:Atom>
    <ruleml:Rel>record complaint cause</ruleml:Rel>
  </ruleml:Atom>
</taskEffects>
```

Given an annotated process and the formalisation of the relevant regulation in LegalRuleML as we shown above, we can use the algorithm proposed in [14] to determine whether the annotated process model is compliant. Shortly, the procedure runs as follows:

- Generate an execution trace of the process.
- Traverse the trace:
  - for each task in the trace, cumulate the effects of the task using an update semantics (i.e., if an effect in the current task conflicts with previous annotation, update using the effects of the current tasks).

- use the set of cumulated effects to determine which obligations enter into force at the current tasks, by calling a reasoner.
- add the obligations obtained from the previous step to the set of obligations carried over from the previous task.
- determine which obligations have been fulfilled, violated, or are pending; and if there are violated obligation check whether they have been compensated.

– repeat for all traces.

A process is evaluated as compliant if and only if all traces are compliant (all obligations have been fulfilled or if violated they have been compensated), or it is evaluated as weakly compliant if there is at least one trace that is compliant.

Soundness and completeness of the proposed methodology depend on the data (rules and semantic annotations) associated with a business process. The methodology is sound and complete provided that the rules are an appropriate interpretation of the norms, and the semantic annotations are complete. If this is the case the computational model supported by PCL properly simulates legal reasoning. Otherwise, there are two possible issues. The process is not compliant because some semantic annotation is missing. This is the case of unfulfilled obligation, that is, there is some obligation [OANPP]p that is force in some tasks (trace) but we do not have evidence for p in the tasks (trace). In this case Regorous report such issue and the user can add the information to some tasks if appropriate. For the other case, it is possible to avoid some obligations by failing to trigger some rules. For example, given the rule $p \Rightarrow$ [OANPP]q, we can avoid the obligation of q if p is not an effect of some task. To handle this situation, Regorous asks for justification for the rules that are not used in the process (and it is up the user again to provide the information if appropriate, p could be facultative, and there is no need to have it).

4 Evaluation

In this section, we first present the evaluation of our approach (Section 4.1), and then we discuss the lessons learned (Section 4.2).

4.1 Results

The approach proposed in this paper has been evaluated in a six week pilot project in collaboration with an industry partner (a small to medium Australian telecommunication service provider, about 70,000 customers at the time of the evaluation), and the regulator. For the evaluation, Chapter 8 of the Code on complaint handling was selected. A legal knowledge engineer from our group manually mapped Chapter 8 of the 2012 version of code in LegalRuleML, and XSLT transformations are used to translate the LegalRuleML representation in PCL as used by Regorous. The mapping of Chapter 8 took approximately 2 weeks. The chapter contains approximately 100 paragraphs, in addition to

approximately 120 terms given in the Definitions and Interpretation section of the Code (Chapter 2). The mapping resulted in 176 LegalRuleML normative statements, containing 223 distinct RuleML atoms (`<ruleml:Atom>`), and 7 LegalRuleML overrides statements (`<lrml:overrides>`). Of the 176 normative statements, 33 were constitutive statements (`<lrml:ConstitutiveStatment>`) used to capture definitions of terms used in the remaining rules. Mapping the section required all features of PCL. The regulator examined the mapping, and they deemed it to be a suitable interpretation of the Code.

For the second phase of the evaluation, we had a series of 1-day workshops with the industry partner. The industry partner did not have formalized business processes. Thus, we worked with domain experts from the industry partner (who had not been previously exposed to BPM technology, but who were familiar with the industry code) to draw process models for the activities covered by the Code. The evaluation was carried out in two steps. In the first part, we modelled the processes as they were. It took two workshops to teach them how to model business processes, and to jointly model their existing processes related to complaint handling and managements of complaints and complaints procedures. The third 1-day workshop was dedicated to add the semantic annotation to the business processes. The domain experts were able to complete the task in one afternoon after they were instructed on how to do in the morning.

Regorous was able to identify several areas where the existing processes were not compliant with the new code. In some cases the industry partner was already aware of some of the areas requiring modifications of the existing processes given that the Code introduced totally new requirements (for example the need to address in person or by phone complaint immediately). However, some of the compliance issues discovered by the tools were novel to the business analysts and were identified as genuine non-compliance issues to be resolved. Some of these issues where due to subtle changes in the Code while other were discover by the in deep analysis forced by the methodology implemented by Regorous which would be hard to detect with manual analysis. In the final part of the experiment, the existing processes were modified to comply with the Code based on the issues identified in the first phase. In addition, a few new business process models required by the new code were designed. As result, we generated and annotated 6 process models. 5 of the 6 models are limited in size and they can be checked for compliance in seconds. The largest process contains 41 tasks, 12 decision points, xor splits, (11 binary, 1 ternary). The shortest path in the model has 6 tasks, while the longest path consists of 33 tasks (with 2 loops), and the longest path without loop is 22 task long. It takes approximately 40 seconds to verify compliance for this process on a MacBook Pro 2.2Ghz Intel Core i7 processor with 8GB of RAM (limited to 4GB in Eclipse).

Due to a confidentiality agreement with the industry partner it is not possible to release the process models used in the evaluation. However, the Regorous compliance checker is available under a free evaluation license at `http://www.regorous.com`. The distribution includes some simple but realistic scenarios consisting of business process models and fragments of relevant regulations. The

scenarios can evaluate the compliance of simple processes (10–15 tasks, a few decision nodes, and about 30–40 rules) in matter of seconds running in a standard laptop with the same specifications as the computer used for the evaluation. Note that the complexity of checking the compliance of a business process is in function of the complexity of the underlying business process model (and linear in the number of rules and propositions). The problem has been shown to be NP-complete even when the correctness of processes is in PTIME, and the legal reasoning in the single tasks of a process is in PTIME as well [4]. Given that the complexity of the reasoning tasks in PCL is in PTIME [13], the complexity of a business process depends on the number of states traversed by the traces of a business processes, which is potentially exponential in the number of tasks appearing in a process in function of the control flows nodes (connectors). The most complex process in the pilot study consists of approximately 24,000 states and processes included in the Regorous distribution are between 50 and 250 states. Thus, while the rate states/response time for the samples scenario is one order of magnitude larger than that of the pilot case, the response time for the process in the pilot case can be also extrapolated from the response time from the other processes based on the theoretical results on complexity when one accounts for initialisation time and some optimisations in the implementation that allows us to avoid the computation in states that are already computed. Specifically, the set of traces corresponding to a process is represented as a tree, thus the states that are common to multiple traces are computed only once. Thus, for example, in a process of 10 tasks in sequence followed by an XOR spilt with two branches, the 10 initial states are common to the two branches, and it is pointless to compute them twice.

4.2 Lessons learned and future work

The results presented in the previous section demonstrate the effectiveness of the semantically enriched business process regulatory compliance checking mechanism we proposed. Besides these considerations, some further positive and negative insight emerged during the evaluation we conducted with the industry partner:

Positive feedback: First, we discovered, together with the domain experts of the industry partner, that exploiting a semantic model such as LegalRuleML allows us to embed much more information in the rules representing the regulatory code. This enhancement in the precision of the legal provisions leads to an enhancement of the regulatory checking phase, as we compared two more fine-grained representations of the Code and of the tasks, respectively. Second, the semantics of LegalRuleML allows for the evolution over time of the regulations to be compliant with. This has the advantage of tracking when a change occurred, and what is the context to be used depending if the compliance is verified against the new or the old version of the Code. This is also a valuable benefit of the adoption of this semantic model with respect to simple rule-based formats.

Negative feedback: Even if the semantic model allows to provide a more faithful representation of the legal document, it is not straightforward to understand the semantic model of LegalRuleML and how it works. It required some time to the domain experts of the industry partner to understand how to match the rules present in the Code they were aware of, and the LegalRuleML semantics. As future work, we need to develop a graphical interface to interact with such a complex model so that examples of rules in natural language from existing regulations are provided and translated into LegalRuleML so that the experts are supported in their modeling phase. In summary, the result of our experience with the industry partner showed us that users are much better in using semantically enriched documents rather than in creating them. Finally, the showstopper is that the extraction of the rules from the legal documents is time consuming and there is a huge need to support the translation of such legal documents from natural language to their `xml` counterpart. This is another line we will address as future work.

5 Related Work

The problem of providing a machine-readable semantic representation of legal knowledge has been addressed in different domains, leading to the definition of various ontologies targeting different legal contexts. Among others, there are the Open Digital Rights Language (ODRL) for rights expressions[10], the Functional Ontology for Law [25] about normative knowledge, world knowledge, and responsibility knowledge, the Frame-Based Ontology of Law [19] about norms, acts and concepts descriptions, the IKF-IF-LEX Ontology for Norm Comparison [8] about agents, institutive and regulative norms, and norm dynamics, the LKIF-Core Ontology[11] including the OWL ontology of fundamental legal concepts [22]. However, all these works differ from LegalRuleML for what concerns *(i)* the use of rules to account for the specifics of the legal domain, and *(ii)* the use of a legal reasoning level on top of the ontological layer of the Semantic Web stack. LegalRuleML allows users to specify in different ways how legal documents evolve, and to keep track of these evolutions and connect them to each other, as we exploited in this paper.

To our knowledge, there is no other approach addressing the problem of a semantic business process regulatory compliance: in this work, we not only exploited Semantic Web technologies and languages to propose different modelling techniques to represent the legal information contained in the legal documents and their dynamics, i.e., the Code (2012 and 2016), but we empowered a business process compliance system with a semantic annotation of the rules and the processes. An approach to semantic business process compliance management has been proposed by El Kharbili and colleagues [5]. We share the idea of making use of the advantages of semantic technologies for compliance management. However, the two approaches are different. First of all, we are interested in regulatory

[10] `https://www.w3.org/ns/odrl/2/ODRL21`
[11] `http://www.estrellaproject.org/lkif-core/`

compliance, and not in business process management in general, which means that we need to exploit the powerful semantics of the LegalRuleML framework to convey the semantics of the rules extracted from the legal documents. We do not need to design a business policy and business rule ontology, as in [5]. Second, the proposed architecture considers also the dynamics of the legal documents to be checked the compliance with, by proposing alternative modeling solutions. Finally, we annotate the tasks included in the processes with the semantic of LegalRuleML, so that automated compliance checking is done at semantic level. Besides Regorous a few other compliance prototypes have been proposed. Here we consider some representative ones: Compass [7] and SeaFlows [21]. However, none of them exploits Semantic Web languages and technologies, and they are not compliant with the guidelines set up in [9] for rule languages for the representation of legal knowledge and legal reasoning. In addition, such approaches have severe limitations in modelling legal reasoning, since they do not provide a conceptually sound model of legal reasoning [12].

6 Conclusions

In this paper, we have presented a semantic approach to business process regulatory compliance checking. Regulatory compliance checking is a major challenge for companies and institutes, and being supported by automated techniques in such a verification phase results in a valuable gain of time and money. We have reported here our experience in applying Semantic Web technologies and languages to this challenging task in the context of a project with an industry partner. Accounting for the complexity and the required precision in modelling norms and regulations, and in checking whether a certain process and its related tasks are actually compliant with the normative system they are subject to, we propose a semantic approach based on the LegalRuleML semantic model. Our evaluation shows that our system is able to capture the semantics of the Code and to model its dynamics in a satisfactory way, and to efficiently check the compliance of processes with respect to this reference Code. The lessons learned during this project will guide our future work, that includes also the evaluation of the Regorous semantic system with larger processes, and applying this methodology with other kinds of regulations in order to make Regorous more flexible to the needs of the companies adopting it.

References

1. Athan, T., Boley, H., Governatori, G., Palmirani, M., Paschke, A., Wyner, A.: OASIS LegalRuleML. In: Francesconi, E., Verheij, B. (eds.) ICAIL 2013. pp. 3–12. ACM (2013)
2. Athan, T., Governatori, G., Paschke, A., Palmirani, M., Wyner, A.: LegalRuleML: Design principles and foundations. In: Faber, W., Paschke, A. (eds.) Reasoning Web. Web Logic Rules, pp. 151–188. Springer (2015)
3. Bench-Capon, T., Coenen, F.P.: Isomorphism and legal knowledge based systems. Artificial Intelligence and Law 1(1), 65–86 (1992)

4. Colombo Tosatto, S., Governatori, G., Kelsen, P.: Business process regulatory compliance is hard. IEEE Transactions on Services Computing 8(6), 958–970 (2015)
5. Decreus, K., Poels, G., El Kharbili, M., Pulvermüller, E.: Policy-enabled goal-oriented requirements engineering for semantic business process management. Int. J. Intell. Syst. 25(8), 784–812 (2010)
6. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. Springer (2013)
7. Elgammal, A., Türetken, O., van den Heuvel, W.J.: Using patterns for the analysis and resolution of compliance violations. International Journal of Cooperative Information Systems 21(1), 31–54 (2012)
8. Gangemi, A., Breuker, J.: Harmonizing legal ontologies. In: Deliverable 3,4 IST Project-2000-29243. Ontoweb (2002)
9. Gordon, T.F., Governatori, G., Rotolo, A.: Ruleml 2009. In: Governatori, G., Hall, J., Paschke, A. (eds.) RuleML 2009. pp. 282–296. Springer, Berlin (2009)
10. Governatori, G.: Representing business contracts in RuleML. International Journal of Cooperative Information Systems 14(2-3), 181–216 (2005)
11. Governatori, G.: The Regorous approach to process compliance. In: EDOC 2015 Workshop. pp. 33–40. IEEE (2015)
12. Governatori, G., Hashmi, M.: No time for compliance. In: Hallé, S., Mayer, W. (eds.) EDOC 2015. pp. 9–18. IEEE (2015)
13. Governatori, G., Rotolo, A.: BIO logical agents: Norms, beliefs, intentions in defeasible logic. Journal of Autonomous Agents and Multi Agent Systems 17(1), 36–69 (2008)
14. Governatori, G., Rotolo, A.: A conceptually rich model of business process compliance. In: Link, S., Ghose, A. (eds.) APCCM 2010. pp. 3–12. ACS (2010)
15. Governatori, G., Sadiq, S.: The journey to business process compliance. In: Cardoso, J., van der Aalst, W. (eds.) Handbook of Research on BPM, pp. 426–454. IGI Global (2009)
16. Governatori, G., Shek, S.: Regorous: a business process compliance checker. In: ICAIL 2013. pp. 245–246 (2013)
17. Hashmi, M., Governatori, G., Wynn, M.T.: Business process data compliance. In: RuleML 2012. pp. 32–46. Springer (2012)
18. Hashmi, M., Governatori, G., Wynn, M.T.: Normative requirements for regulatory compliance: An abstract formal framework. Information Systems Frontiers 18(3), 429–455 (2016)
19. van Kralingen, R.: A conceptual frame-based ontology for the law. In: Proc. of the 1st International Workshop on Legal Ontologies. pp. 6–17 (1997)
20. Lam, H.P., Hashmi, M., Scofield, B.: Enabling Reasoning with LegalRuleML. In: Alferes, J.J., Bertossi, L., Governatori, G., Fodor, P., Roman, D. (eds.) RuleML 2016. pp. 241–257. Springer (2016)
21. Ly, L.T., Rinderle-Ma, S., Göser, K., Dadam, P.: On enabling integrated process compliance with semantic constraints in process management systems - requirements, challenges, solutions. Information Systems Frontiers 14(2), 195–219 (2012)
22. Rubino, R., Rotolo, A., Sartor, G.: An OWL ontology of fundamental legal concepts. In: van Engers, T.M. (ed.) JURIX 2006. pp. 101–110. IOS Press (2006)
23. Sadiq, S., Governatori, G., Naimiri, K.: Modelling of control objectives for business process compliance. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. pp. 149–164. Springer, Berlin (2007)
24. Searle, J.: The Construction of Social Reality. The Free Press (1996)
25. Valente, A., Breuker, J.: A functional ontology of law. In: Artif. Intelligence and Law. pp. 7:341–361 (1994)